De-anonymizing D4D Datasets

Kumar Sharad[1] and George Danezis[2]

[1] University of Cambridge Computer Laboratory,
15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
`Kumar.Sharad@cl.cam.ac.uk`
[2] Microsoft Research
21 Station Road, Cambridge CB1 2FB, UK
`gdane@microsoft.com`

Abstract. Recent research on de-anonymizing datasets of anonymized personal records has not deterred organizations from releasing personal data, often with ingenuous attempts at defeating de-anonymization. Studying such techniques provides scientific evidence as to why anonymization of high dimensional databases is hard and throws light on what kinds of techniques to avoid. We study how to de-anonymize datasets released as a part of *Data for Development* (D4D) challenge [12]. We show that the anonymization strategy used is weak and allows an attacker to re-identify and link records efficiently, we also suggest some measures to make such attacks harder.

1 Introduction

As we continue to digitize our lives it is becoming progressively easier to document our behavior. In today's world each of us have bank transaction histories, call detail records, shopping histories, etc. maintained by various parties. Researchers such as sociologists and data scientists are specially interested in studying such data. Consequently, such data is released by organizations to conduct scientific studies.

However, this presents the problem of privacy intrusion of individuals. Organizations releasing private data attempt to solve this problem by anonymizing the data and to make re-identification of data impossible. The question whether anonymization is sufficient for privacy has seen active debate recently, with studies suggesting approaches to anonymize and de-anonymize data. Often sensitive data is released for research which leads to privacy breaches of various kinds. Research has shown repeatedly that anonymizing feature rich data is extremely hard and in practice such attempts do not work, some examples of such work are [11, 9, 10, 15, 2] and [7]. Techniques have also been developed to protect anonymized data, some such examples are [4, 16] and [14]. However, Dwork and Naor [3] have shown that preserving privacy of an individual whose data is released cannot be achieved in general.

Social networks are a very good example of high dimensional databases and they have information densely packed into them. At the same time it is very challenging to anonymize them while still maintaining the usefulness of the data. Often anonymization techniques make assumptions about the side-information that do not hold. Organizations have released social network databases and techniques developed have been successful in defeating the anonymization strategies employed [11, 9, 10].

Due to the challenges faced in protecting privacy in the case of social network data release, one needs to carefully study any such scheme which attempts to protect privacy, since in general it is not possible. In this paper we evaluate such a scheme on behalf of a mobile network operator (Orange). In July 2012 Orange introduced the *Data for Development* (D4D) challenge [12] as an open data challenge to encourage research teams around the world to analyze datasets of anonymous call patterns collected at Orange's Ivory Coast subsidiary. The motivation behind this challenge was to help address the questions regarding development in novel ways. The mobile network operator wanted to ensure that the data being released does not jeopardize the privacy of the individuals even after proper anonymization procedures being deployed. To evaluate this attempt a preliminary dataset was made available to us after signing an appropriate non-disclosure agreement. We examined the datasets and advised the mobile network operator accordingly. After considering our suggestions the datasets were modified prior to release. The details of the datasets made available to us can be found in section 4.

In total four datasets were released for analysis, in this paper we study the *Dataset 4* – motivation behind releasing this dataset was to allow researchers to study social interactions by analyzing communication graphs. This dataset contains the communication sub-graphs of about 8300 randomly selected subscribers, referred to as egos. The sub-graphs provide all the communications between the egos and their contacts up to 2 degrees of separation, the data also includes the number of calls between two users in a ego network and the duration of each call. Communication between the users has been divided into periods of two weeks spanning 150 days.

The individuals were assigned random identifiers which remain same for all the time slots. However, to obfuscate the interactions between ego nets the common members of the ego-graphs of two different customers were provided unique identifiers, i.e. if an individual was a part of ego networks of two different egos then he had a different identifier in each one of them.

It is not obvious how this dataset can be exploited to compromise privacy but due to the unique nature of social networks and interactions between the members we show how this dataset could be a major concern for privacy protection. We present a detailed analysis in section 3.

2 The Problem

The anonymization strategy for *Dataset 4* tries to disconnect the ego nets published so as to conceal the overall graph structure. The knowledge of graph topology can cause severe privacy breach even if only a few nodes are re-identified as rest of the structure can be ascertained from the topology itself. We see that graph topology alone is not a big threat but once the full graph is known a standard technique can be used to re-identify. Before attempting to de-anonymize *Dataset 4* we need to formally describe the problem. We study the problem at hand using an example, the given dataset contains the communication of all the individuals in the ego net graph of an user upto the depth of 2. To illustrate this we use Figure 1 and Figure 2 which are ego nets extracted from a real world social network. These ego nets are centred at the *red* node, *orange* nodes denote 1-hop nodes and *blue* nodes denote 2-hop nodes.

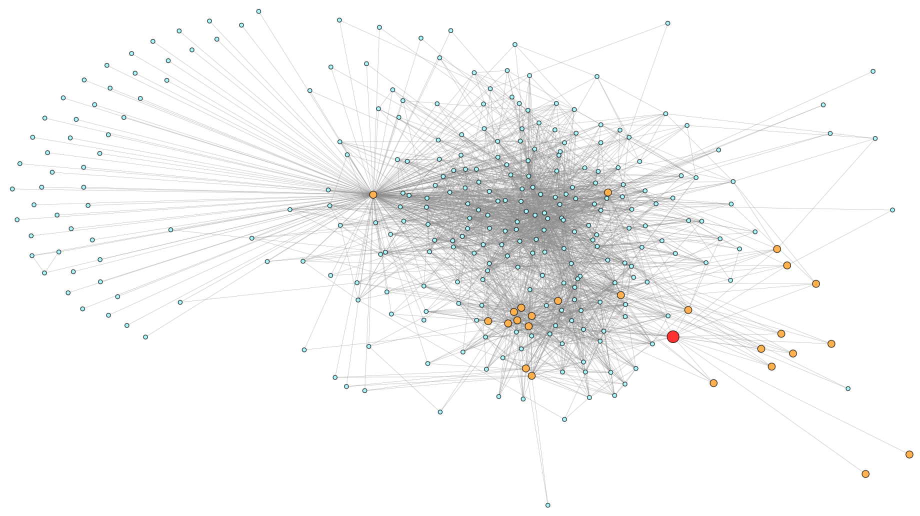

Fig. 1: The ego net G_0

In this example some nodes are common between graphs G_0 and G_1, on constructing node induced graph of the common nodes we discover that they interact in intricate ways as shown in Figure 3. Using this example we wish to illustrate the problem and motivate a solution. *Dataset 4* gives us access to thousands of ego graphs whose labels have been anonymized and are unique across ego nets for different egos, due to this the links between various ego nets have been lost. The statistical properties of social graphs indicate that they tend to be heavily clustered and hence there will be pairs like (G_0, G_1) which have significant overlap compared to the size of the ego nets.

It can be already seen at this point that even if we know that a pair of graphs have overlapping nodes it is not clear how we can map such nodes when the identifiers have been scrambled. All we have at this point is the graph topology and the weights of directed edges. This information can we used to assign an edge weight to every interaction between the nodes, we can say that *node A*

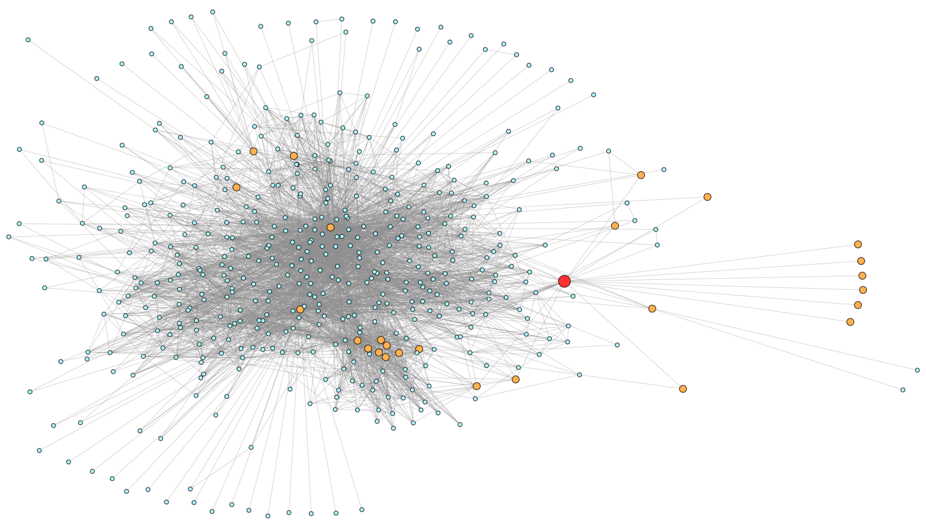

Fig. 2: The ego net G_1

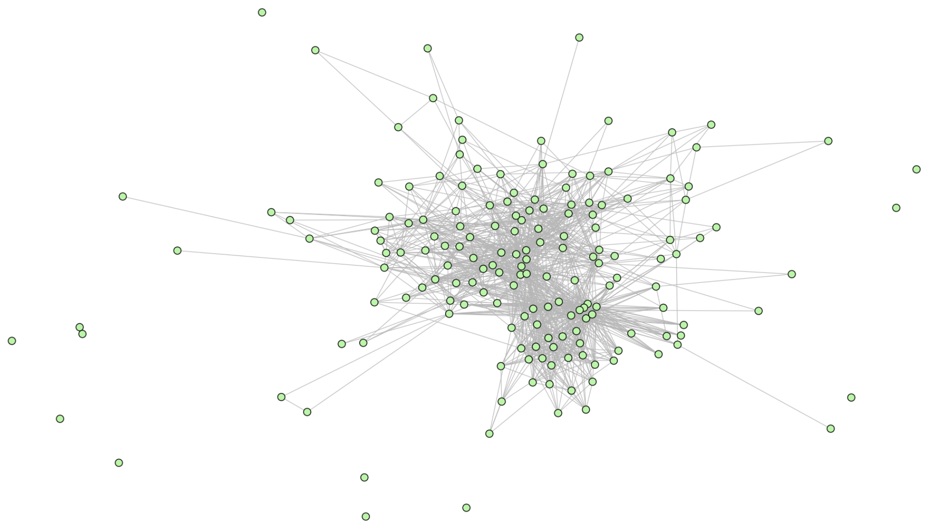

Fig. 3: Sub-graph common to both G_0 and G_1

makes x calls to *node B* that last for a total duration of time y then the weight of the edge between the nodes is (x, y).

Essentially, we are looking for sub-graphs of G_0 and G_1 which are isomorphic and are largest such sub-graphs. If we can find significant overlap between two graphs then the larger the matching sub-graph the higher the likelihood that the match is true. Finding isomorphic graphs of sizes 2 or 3 nodes which are common to any given pair of graphs is quite probable. Finding a false positive large match between ego nets of a social network is extremely rare.

Ideally we would like to map all the common anonymized nodes across pairs like (G_0, G_1) and reconstruct the union of graphs G_0 and G_1. In this simple example such a graph would look like the one shown in Figure 4, again the *red* nodes denote the center nodes, the *orange* nodes are at 1-hop distance and the *blue* nodes are at 2-hop distance. We can extend this approach further to many sub-graphs namely $G_0, G_1, \ldots, G_n$ of which several pairs have overlapping nodes then by combining them together we can recover the entire graph from which the sub-graphs were extracted. In the remainder of the paper we investigate how to re-link the ego nets to reveal the structure of the graph and exploit it to divulge identities.

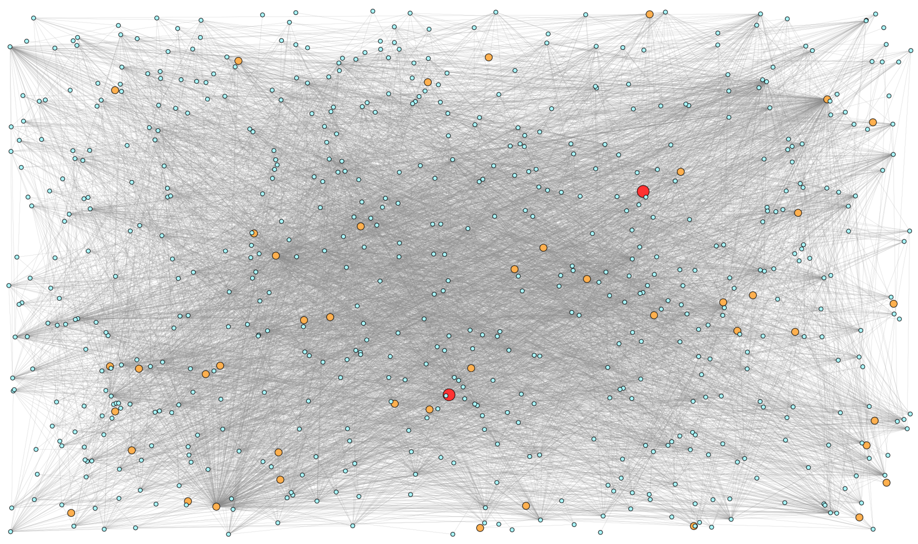

Fig. 4: The complete graph G

3 Proposed Solution

Pedarsani and Grossglauser [13] have shown that it is feasible to de-anonymize a target network by using the structural similarity of a known auxiliary network, we build on their work to develop our attack. Section 2 highlights the problem that needs to be solved to map anonymized nodes across ego nets. One possible approach is finding sub-graphs which are common between a pair of ego nets. Additionally this sub-graph should be the largest such graph. This is referred to as the maximum common subgraph-isomorphism problem and is known to be NP-hard in general, even if one were to assume that we have an instance of the problem that can be solved efficiently it would still be too computationally expensive to launch an attack. Hence we need to explore other approaches to solve this problem.

3.1 Intuition

Our approach involves dividing the problem into parts and solving them individually. We have a 2-hop network of each ego at our disposal. If two graphs have nodes in common then they can be classified under three categories, namely

(a) Both nodes are at a distance of 1-hop from the center.
(b) One node is at a distance of 1-hop while the other node is at a distance of 2-hop from the center.
(c) Both nodes are at a distance of 2-hop from the center.

We have the complete neighborhood graph of nodes falling in the category (a), whereas, the nodes falling in category (b) and (c) have at least a part of their neighborhood information deleted because of being paired with a 2-hop node. Hence, it should be easier to re-identify nodes which fall under the category (a) as compared to nodes which fall under the categories (b) and (c).

As mentioned earlier, due to problem being NP-hard in general we exploit the structure of the problem that allows for an efficient solution. It is important to note that our techniques make use of structural properties of social graphs, and they are only applicable to other problems if similar properties are present. Pursuing this argument further, we utilize the degree distribution of nodes of social networks which tend to be clustered and follow the power-law [6]. This indicates that if one were to observe the neighborhood of a particular 1-hop node and list the degree of all its neighbors then it is fairly unlikely that another node of the graph will exhibit the same distribution, provided the node degree is not extremely low. This approach is detailed in section 3.2.

For the nodes which fall under the categories (b) and (c) this approach will not work because portions of the neighborhood of the 2-hop nodes have been deleted. We take a different approach to match such nodes. Broadly speaking, the attack works by assigning a match score to each pair of nodes, finally pairs are selected in such a way that maximizes the entire score thus generating a final mapping. Our attack builds on the de-anonymization of nodes falling under category (a) and assumes that a pair of graphs have some 1-hop nodes common. We discuss the details in section 3.3.

3.2 Matching 1-hop Nodes

As outlined earlier we de-anonymize the common 1-hop nodes between two graphs by observing their 1-hop neighbourhood degree distribution. We proceed by scanning the 1-hop neighbourhood of a particular node and collecting the node degrees. These are then sorted in a list and we treat it as a signature of that particular 1-hop node. After collecting the signatures of all the 1-hop nodes of a pair of graphs we proceed to match the signatures between graphs. If a match is found then we consider the corresponding nodes to be same. The key thing to note here is that we have the complete 1-hop neighbourhood of a particular 1-hop node and this makes it possible to implement this attack. Social network graphs represent real people and their behaviours, hence, finding exactly same degree distribution of two nodes with sufficiently high number of neighbours is highly unlikely. Algorithm 1 illustrates our attack.

Algorithm 1 Matching 1-hop nodes between two ego nets

```
Input: Ego nets $G_0$ and $G_1$
Output: Mapping between 1-hop nodes of $G_0$ and $G_1$

 1: function GRAPH_CENTER($G$)
 2:     return the ego of the graph

 3: function SIGNATURE($G$)
 4:     center ← GRAPH_CENTER($G$)
 5:     List all the neighbours of center in ego_neighbours
 6:     for node in ego_neighbours do
 7:         Create 1-hop graph centred at node called G_temp
 8:         for all nodes in G_temp do
 9:             Append degree in list node_sig
10:         Append node_sig keyed by node in dictionary G_dict
11:     Invert G_dict keying nodes by signature
12:     return G_dict

13: $G_0$_sig ← SIGNATURE($G_0$)
14: $G_1$_sig ← SIGNATURE($G_1$)

15: for signature in $G_0$_sig keys do
16:     if $G_1$_sig contains signature then
17:         Map signature value of $G_0$_sig and $G_1$_sig in list mapping
18: Output mapping
```

Dataset 4 also gave us access to frequency and the duration of calls between the users. This can be used to relate them using edge weights as shown in section 2, thus creating a weighted graph. Having edge weights in the graph allows us to improve the efficiency of the algorithm by first determining the common edge weights between the 1-hop nodes of the two ego nets and then computing the signature of only those nodes which are connected by edges with these weights. The improvement depends upon the ratio of the total edges present in 1-hop graph to the unique edge weights present.

3.3 Matching the non-1-hop Nodes

To match the nodes which fall in the category (b) and (c) we build on the progress made in section 3.2. It is observed that a pair of social network graphs that have 1-hop nodes in common tend to have a significant number of 2-hop nodes in common as well. We cannot follow an approach similar to mapping 1-hop nodes to map the 2-hop nodes since major portion of their neighbourhood might be missing, hence using it as a basis to identify the nodes will give erroneous results. However, we must make use of the progress made in mapping 1-hop nodes to achieve success.
We assume at the outset that the pair of ego nets whose 2-hop nodes we are trying to map already have common 1-hop nodes, these nodes can be identified by using Algorithm 1. After this step we compute the neighbor match as a cosine distance between a particular pair of 2-hop nodes of G_0 and G_1 by observing which of the common 1-hop nodes they are connected to. If the distance is 1 then we proceed to compute the signature match. The signature is defined similarly as we did in the case of 1-hop nodes except with one difference – whereas earlier we considered the full degree of each of the node neighbors here the degree of a node is represented as a tuple of in and out degrees. This is done to compute a more accurate signature match, earlier we had the complete neighborhood graph so to identify two nodes as being the same all we needed to do was to look for exact signature matches, however, we cannot afford to do this here hence we compute a cosine distance between signatures. Thus representing the signature by using a tuple of in and out degree is more accurate than adding them up and gives a higher confidence in computed distance between signatures. The final match is computed as the average of neighbor match and signature match, this match score is used as the edge weight of a bipartite graph whose nodes are the nodes of G_0 and G_1 respectively. Once we finish populating the bipartite graph we use the Munkres assignment algorithm [8] to find the optimal mapping between nodes of the bipartite graph that maximizes the sum of edge weights.
We only do these computations for nodes that have a significantly high degree. For low degree nodes have little information available to de-anonymize them with high confidence. For our experiments we consider nodes with degree greater than 10. Algorithm 2 in appendix A illustrates our attack. This algorithm can also be improved by considering the edge weights of the entire graph and then using them as discussed in section 3.2.

3.4 Results

We analysed our algorithms by running them on the *EU email communication network* taken from the *Stanford Large Network Dataset Collection* [5]. We recreated a dataset similar to the *Dataset 4* by extracting egonets as done in the original scheme. The data was treated as anonymized before running the algorithm, subsequently the results were compared based on the ground truth. Although, the data used for evaluating our algorithms does not represent call records, still being a social network it has similar properties and represents human behavior.

Our analysis indicates that Algorithm 1 is highly effective in re-identifying almost all the 1-hop nodes common between a pair of egonets with success percentage of over 98. The approach works very well even on nodes of low degree, however, nodes which are completely stranded with just 2 or 3 neighbors could not be identified due to sparsity of information but very few such nodes exist in a 1-hop graph.

The results of Algorithm 2 are illustrated in Table 1. Algorithm 2 was also quite successful with good success percentage in most cases. However, the success rate drops if we try to de-anonymise more nodes. As mentioned earlier we have considered nodes which have degree greater than 10 in our analysis this gives us good rates of in most cases but we pay the penalty of losing over 70% of nodes in most cases. It is typical of social networks to have many nodes of lower degree and our approach works only in cases where degree is over 10. It seems likely that the cost of identifying nodes with degree lower than 10 could be exponential in the size of the nodes. Also, our solution obtains mixed results in cases where the graph contains high percentage of nodes of degree less than 10. The value in brackets in the last two rows of the Table 1 show how the percentages improve when nodes of degree greater than 20 and 12 are selected respectively. This indicates that excluding nodes at a predetermined cutoff could lead to overlap of nodes on either side of the cutoff, for instance there could be certain nodes which have a degree of $k-1$ in one ego net and $k+1$ in another ego net, however excluding all nodes below degree k eliminates the possibility of a good match and introduces errors. On the other hand, if all nodes are allowed then the ones which have low degree obscure the true picture. A methodology to pick up the node degrees appropriately remains to be found.

Table 1 Results of Algorithm 2

G_0 size (nodes)	G_1 size (nodes)	common nodes	re-identification %	Success %
316	503	171	21.64	84.09
2629	2998	307	47.88	96.07
6567	5570	3692	14.65	74.72
8685	5096	1504	36.04	90.48
11690	9264	4325	11.49 (11.12)	69.9 (75.04)
49707	80601	29518	3.98 (3.34)	56.45 (76.37)

4 Dataset Details

As a part of the D4D challenge the mobile network operator released four datasets of anonymized call data records to be studied by researchers. The data sets contained the anonymized *call detail records* extracted from the customer base, spanning a period of 150 days from 1st December, 2011 to 28th April, 2012. The datasets contain the records of individuals of Ivory Coast which has a population of 20 million out of which close to 15 million [1] people use mobile phones. The mobile network operator has 5 million subscribers in Ivory Coast and controls a significant chunk of the market. We take a deeper look at the structure of the datasets and the information provided.

- **Dataset 1** - The first dataset contains the antenna-to-antenna details of the aggregate communications between the cell towers. For every pair of cell towers in Ivory Coast the number of calls and total communication time was provided. The calls were grouped by hourly basis and showed which cell tower initiated the call. Each cell tower was assigned an identifier and the GPS co-ordinates of each tower was also provided. This dataset spans the entire duration of 150 days.

- **Dataset 2** - This dataset provides the cell tower identifiers used by a group of randomly chosen active users to make phone calls and send texts. The data is timestamped and a particular group of users was observed for a period of 2 weeks. At the end of the two week period a fresh sample of active users was drawn at random, each sample contains 10% of the subscriber base and data for 10 such random samples has been provided.
  The phone numbers in each sample are scrambled and assigned random identifiers to protect privacy. It is readily apparent that an individual can be tracked using this dataset as it allows us to build a profile by approximating the geographic location of the user by observing the cell tower being used during a call. By aggregating the entire call history we can obtain a movement profile which reveals the user's behavior pattern. It must be noted that just obtaining the geographic location of the cell tower being used by the user does not give away his location as the tower chosen for communication depends upon several factors like the topology of surrounding towers, traffic, time of the day, etc. Also due to the fact that only parts of Ivory Coast are well developed the network infrastructure and population tends to be clustered, hence there could be many people who correspond to the same antenna location. These details alone are not sufficient for identification, however, as shown by Narayanan and Shmatikov [11] when such data is collated with auxiliary information then this could lead to re-identification. If we are able to identify only a few individuals then this could lead to a snowballing effect and we can compromise more and more people by observing interactions between people already identified and the ones that are anonymous.
  The mobile network operator wanted to provide long term observation data of users as well as high spatial resolution trajectories of individuals due to

the interesting scientific properties such data tends to have. However, the privacy of the individuals needed to be protected along with facilitating scientific study. The solution employed to make identification harder was to publish one dataset with high spatial resolution data for a limited period of time – referred to as *Dataset 2* and another dataset for the entire observation period but with reduced spatial resolution – referred to as *Dataset 3* which we describe next.

- **Dataset 3** - This dataset is similar to the *Dataset 2* apart from the fact that it documents only 1% of the subscriber base, however, the observation period lasts the entire span of 150 days. One further difference between *Dataset 2* and *Dataset 3* is that, to reduce the spatial resolution, instead of providing the cell tower identifiers only the identifier of the sub-prefecture of antenna location is provided. Sub-prefectures are geographical regions of Ivory coast and there are 255 such regions in the country. GPS co-ordinates of the geographic centres of all the sub-prefectures have also been provided.

5 Related Work

In this paper we have concentrated on analyzing the *Dataset 4*, however, for the sake of completeness we find it necessary to mention a few words about the privacy afforded to the users by the strategy chosen to anonymize the other three datasets. Bearing in mind that proper anonymization of high dimensional data is a very hard task, below we discuss the pitfalls and contentious issues that such anonymization raises.

- There is little scope of privacy breach being caused by *Dataset 1* alone since it contains no personally identifiable information about the users. It could be used to study traffic patterns during the entire period but reveals no information pertaining to the users.

- The *Dataset 2* provides the call details of individuals round the clock in each sample. As discussed earlier just by observing the antennas used during communication we cannot obtain the geographic location. But our estimate can be substantially improved by looking at the communication patterns during odd hours for instance during the night, during such periods due to low network traffic chances of an user communicating via the closest tower is significantly higher, also during such hours most people are at their homes. If we observe the call patterns during night and collate it with call patterns during the rest of the day we can decrease the search space for pinpointing an individual. Therefore, to remove this advantage we believe that it is best not to reveal call data for odd hours.

- In *Dataset 2* we observed that some users who exhibit extremely high call volume were communicating using cell towers located in areas that have a very sparse population, for instance villages. Their coarse location can be estimated by observing the most frequently used antenna and in many cases such users used only one antenna. Also places like villages tend to have only a single antenna in their vicinity which makes it even easier to guess the location. The population of Ivory Coast does not have a strong online presence hence it is not possible to identify such people using freely available data online. However, a resident of the area can easily identify such users because in many cases there are only a few candidates in a particular area who can generate such high traffic. Additionally, upon identification (at least with some surety) if this individual can be persuaded sell his call history then this can be combined with the dataset to mount very potent attacks which can de-anonymize a large number of people and cause significant breach of privacy.

- For *Dataset 2*, in general it is possible to identify the group of users to which a particular high volume customer belongs using the local information like addressees of businesses and co-relating it with traffic information.

- It is also possible to identify an individual in *Dataset 2* by using his movement records and a pre-existing location profile. This attack was presented by Mulder et al. [7]. The authors show that they were able to identify around 80% of the users using their approach.

- A given sample of *Dataset 2* contains 10% of the users and 10 such samples are drawn for the entire period of 150 days. So the probability of an individual to appear in a particular dataset is 0.1, conversely the probability that the individual is not present in a given dataset is 0.9. In general it can be stated that $N \times p^{-n}$ people are common across any n samples where N is the total size of subscriber base and p is the probability that an individual would be picked at random in a dataset. Thus calculating the probability that a person is present in atleast one of the datasets $= 1 - (0.9)^{10} = 0.6513$. This implies that *Dataset 2* contains the records of over 65% of the subscriber base which means that every second user's data has been published.

- *Dataset 3* contains just 1% of the subscriber base and provides spatial resolution only up to the sub-prefecture level, hence it is hard to obtain details like coarse location of high volume users and learn from call patterns during the night since we would never be able to go beyond the resolution of sub-prefectures. On the flip side compromising an individual in this dataset has rich rewards since we will obtain the call history for the entire duration.

6 Conclusion and Future Work

In this paper we have shown how nodes of ego nets can be re-identified using graph topology. There is further scope to amplify these attacks and use them to construct an even more serious privacy breach. Though we have partially solved the problem of re-identification, aspects of it remain unsolved and further investigation is required. The techniques presented in this paper are only pertinent to *Dataset 4* and cannot be extended to other datasets. Studying possible privacy breaches caused by other datasets requires a fresh approach. We conclude with a discussion on important considerations for data release and the challenges that lie ahead.

We have shown that releasing a 2-hop ego net allowed us to re-identify almost all the 1-hop nodes. In general if a n-hop network is released then all the $(n-1)$-hop nodes can be re-identified with high probability using the technique presented. Releasing graph data and preserving privacy is an inherently difficult problem due to the enormous amount of information encoded in the graph. To make it harder for the attackers to cause privacy breach methods have been suggested to manipulate graphs. For example in [11] Narayanan and Shmatikov have shown a plausible way of inserting edges in graphs which make attacks harder but preserve the overall graph properties. Future work could evaluate how such strategies impact the success of proposed algorithms, which in turn can be used to refine the attacks.

As seen in previous sections it is possible to re-identify certain class of nodes. Our solution relies on the existence of common 1-hop neighbors. We have found it hard to re-identify nodes when two ego nets have only 2-hop neighbors common, in this scenario our de-anonymization techniques do not work. Of course due the properties of social networks one could attempt to solve this by finding maximum common subgraph-isomorphism between ego nets but this is extremely expensive even for graphs of moderate sizes and is not practical.

Recovering the entire graph remains a challenge as well because firstly, we do not have full confidence in the mapping generated and secondly, we do not have a mapping for all nodes. This introduces inaccuracies while reconstructing the entire graph. However, a sparse representation of the original graph can be constructed by considering the mapping between 1-hop ego nets. Additionally, as discussed in section 3.4 choosing a cut-off appropriately to remove nodes of low degrees is not very clear as it varies depending upon the properties of the ego net. Despite these challenges one should not consider such data anonymization techniques to safeguard the privacy of individuals.

Bibliography

[1] Central Intelligence Agency. CIA - The World Factbook. `https://www.cia.gov/library/publications/the-world-factbook/geos/iv.html`, 2012.

[2] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web*, pages 181–190. ACM, 2007.

[3] C. Dwork and M. Naor. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *Journal of Privacy and Confidentiality*, 2(1):8, 2008.

[4] Michael Hay, Gerome Miklau, David Jensen, Don Towsley, and Philipp Weis. Resisting structural re-identification in anonymized social networks. *Proc. VLDB Endow.*, 1(1):102–114, August 2008. ISSN 2150-8097. URL `http://dl.acm.org/citation.cfm?id=1453856.1453873`.

[5] J. Leskovec, J. Kleinberg, and C. Faloutsos. EU email communication network. `http://snap.stanford.edu/data/email-EuAll.html`, 2007.

[6] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-908-1. doi: 10.1145/1298306.1298311. URL `http://portal.acm.org/citation.cfm?id=1298306.1298311`.

[7] Yoni De Mulder, George Danezis, Lejla Batina, and Bart Preneel. Identification via location-profiling in GSM networks. In Vijay Atluri and Marianne Winslett, editors, *WPES*, pages 23–32. ACM, 2008. ISBN 978-1-60558-289-4. URL `http://dblp.uni-trier.de/db/conf/wpes/wpes2008.html#MulderDBP08;http://doi.acm.org/10.1145/1456403.1456409;http://www.bibsonomy.org/bibtex/2987cab548042306901fe5719aa38f374/dblp`.

[8] James R. Munkres. Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, March 1957.

[9] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 111–125. IEEE, 2008.

[10] A. Narayanan, E. Shi, and B.I.P. Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1825–1834. IEEE, 2011.

[11] Arvind Narayanan and Vitaly Shmatikov. How To Break Anonymity of the Netflix Prize Dataset. *CoRR*, abs/cs/0610105, 2006. URL `http://dblp.uni-trier.de/db/journals/corr/corr0610.html#abs-cs-0610105`.

[12] Orange. D4D Challenge. `http://www.d4d.orange.com`, 2012.

[13] P. Pedarsani and M. Grossglauser. On the privacy of anonymized networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1235–1243. ACM, 2011.

[14] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B.Y. Zhao. Sharing graphs using differentially private graph models. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 81–98. ACM, 2011.

[15] Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A practical attack to de-anonymize social network users. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 223–238. IEEE, 2010.

[16] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 506–515. IEEE, 2008.

A Appendix

Algorithm 2 Matching non-1-hop nodes between two ego nets

```
Input: Ego nets $G_0$, $G_1$ and common 1-hop nodes between them
Output: Mapping between the non-1-hop nodes of $G_0$ and $G_1$

 1: function GRAPH_CENTER($G$)
 2:     return the ego of the graph

 3: function SIGNATURE($G$, $target\_nodes$)
 4:     for $node$ in $target\_nodes$ do
 5:         Create 1-hop graph centred at $node$ called $G\_temp$
 6:         for all nodes in $G\_temp$ do
 7:             Append the tuple of in and out degrees in list $node\_sig$
 8:         Append $node\_sig$ keyed by $node$ in dictionary $G\_dict$
 9:     return $G\_dict$

10: Let $1\_hop\_com$ be the list of all the common 1-hop nodes

11: Let $1hop\_rem\_1$ be the list of 1-hop nodes in $G_0$ not present in $1\_hop\_com$
12: Let $1hop\_rem\_2$ be the list of 1-hop nodes in $G_1$ not present in $1\_hop\_com$

13: Let $neighbors\_2hop\_1$ be a dictionary mapping each of $G_0$'s 2-hop nodes to
    their neighbours in $1\_hop\_com$
14: Let $neighbors\_2hop\_2$ be a dictionary mapping each of $G_1$'s 2-hop nodes to
    their neighbours in $1\_hop\_com$
```

```
15: Let $neighbors\_1hop\_rem\_1$ be a dictionary mapping each node in $1hop\_rem\_1$
    to their neighbours in $1\_hop\_com$
16: Let $neighbors\_1hop\_rem\_2$ be a dictionary mapping each node in $1hop\_rem\_2$
    to their neighbours in $1\_hop\_com$

17: Let $G\_bipar$ be a bipartite graph

18: for $n1$ in $neighbors\_2hop\_1$ do
19:     $s1 \leftarrow neighbors\_2hop\_1[n1]$
20:     for $n2$ in $neighbors\_2hop\_2$ do
21:         $s2 \leftarrow neighbors\_2hop\_2[n2]$
22:         $n\_match \leftarrow$ cosine match of $s1$ and $s2$
23:         if $n\_match = 1$ then
24:             Append $n1$ to list $node\_lst1$
25:             Append $n2$ to list $node\_lst2$
26:             Add $n\_match$ to dictionary $dict\_match\_1$ keyed by $(n1, n2)$

27: $G_0\_2hop\_sig \leftarrow$ SIGNATURE($G_0$, $node\_lst1$)
28: $G_1\_2hop\_sig \leftarrow$ SIGNATURE($G_1$, $node\_lst2$)

29: for $pair$ in $dict\_match\_1$ do
30:     $n1 \leftarrow pair[0]$
31:     $n2 \leftarrow pair[1]$
32:     $sig1 \leftarrow G_0\_2hop\_sig[n1]$
33:     $sig2 \leftarrow G_1\_2hop\_sig[n2]$
34:     $sig\_match \leftarrow$ cosine match between $sig1$ and $sig2$
35:     if $sig\_match > 0$ then
36:         $n\_match \leftarrow dict\_match\_1[pair]$
37:         $final\_match \leftarrow (n\_match + sig\_match)/2$
38:         Insert edge $(n1, n2)$ with weight $final\_match$ in $G\_bipar$

39: for $n1$ in $neighbors\_1hop\_rem\_1$ do
40:     $s1 \leftarrow neighbors\_1hop\_rem\_1[n1]$
41:     for $n2$ in $neighbors\_2hop\_2$ do
42:         $s2 \leftarrow neighbors\_2hop\_2[n2]$
43:         $n\_match \leftarrow$ cosine match of $s1$ and $s2$
44:         if $n\_match = 1$ then
45:             Append $n1$ to list $node\_lst3$
46:             Append $n2$ to list $node\_lst4$
47:             Add $n\_match$ to dictionary $dict\_match\_2$ keyed by $(n1, n2)$

48: for $n1$ in $neighbors\_2hop\_1$ do
49:     $s1 \leftarrow neighbors\_2hop\_1[n1]$
50:     for $n2$ in $neighbors\_1hop\_rem\_2$ do
51:         $s2 \leftarrow neighbors\_1hop\_rem\_2[n2]$
```

```
52:         $n\_match \leftarrow$ cosine match of $s1$ and $s2$
53:         if $n\_match = 1$ then
54:             Append $n1$ to list $node\_lst3$
55:             Append $n2$ to list $node\_lst4$
56:             Add $n\_match$ to dictionary $dict\_match\_2$ keyed by $(n1, n2)$

57: $G_0\_12hop\_sig \leftarrow \textsc{signature}(G_0,\ node\_lst3)$
58: $G_1\_12hop\_sig \leftarrow \textsc{signature}(G_1,\ node\_lst4)$

59: $dict\_final\_1 \leftarrow G_0\_2hop\_sig + G_0\_12hop\_sig$
60: $dict\_final\_2 \leftarrow G_1\_2hop\_sig + G_1\_12hop\_sig$

61: for $pair$ in $dict\_match\_2$ do
62:     $n1 \leftarrow pair[0]$
63:     $n2 \leftarrow pair[1]$
64:     $sig1 \leftarrow dict\_final\_1[n1]$
65:     $sig2 \leftarrow dict\_final\_2[n2]$
66:     $sig\_match \leftarrow$ cosine match between $sig1$ and $sig2$
67:     if $sig\_match > 0$ then
68:         $n\_match \leftarrow dict\_match\_2[pair]$
69:         $final\_match \leftarrow (n\_match + sig\_match)/2$
70:         Insert edge $(n1, n2)$ with weight $final\_match$ in $G\_bipar$

71: Run Munkres assignment algorithm on $G\_bipar$ and output the optimal mapping of nodes between $G_0$ and $G_1$
```